Organization Structures

It seems that remarkably early in our lives we become familiar with organizational structures. The classic management hierarchy appears on an org chart early in our career, but even by then we've already come across the notion in plenty of places. So in a way it shouldn't be surprising that organization structures crop up frequently enough in business software too. I recognized many organizational patterns several years ago and ever since they keep turning up again.

A good way to start thinking about modeling organization structures is to think of the obvious way. Imagine a company where people work in departments, which are organized into divisions. Figure 0.1 shows an explicit model for this where each part of the structure is a separate class.

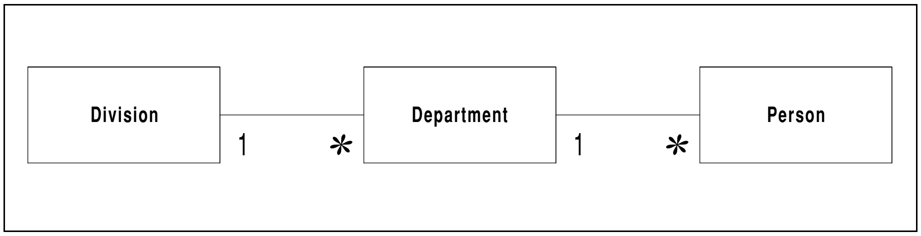

Figure 0.1 *An explicit, and obvious,organizational structure.*

Explicit structures have two main disadvantages. They don't work well if there is much common behavior between the kinds of organization. They also embed the current organizational categories into the design. Should some bright spark decide to add regions between divisions and departments, you have some modificaitons to do.

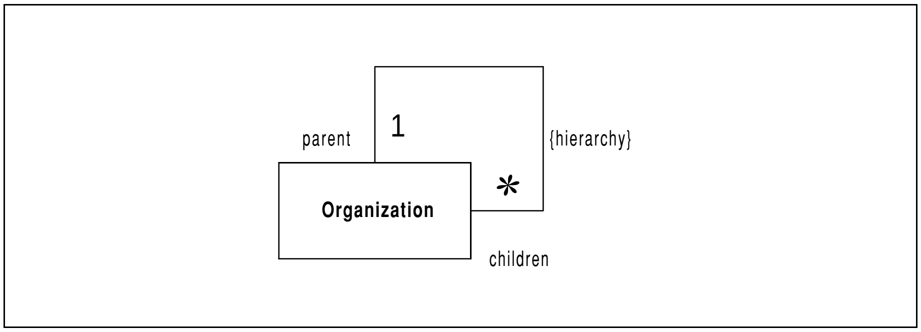

Figure 0.2 *Organization hierarchy*

Faced with these problems, the obvious move is to create a supertype for the organization, which leads you to *Organization Hierarchy (17)* in Figure 0.2. The organization hierarchy works best when you don't have much different behavior between the organization sructures. It also allows you to stay very flexible if new kinds of organizations appear. If you do have some varied behavior you can use subtypes to pull this down.

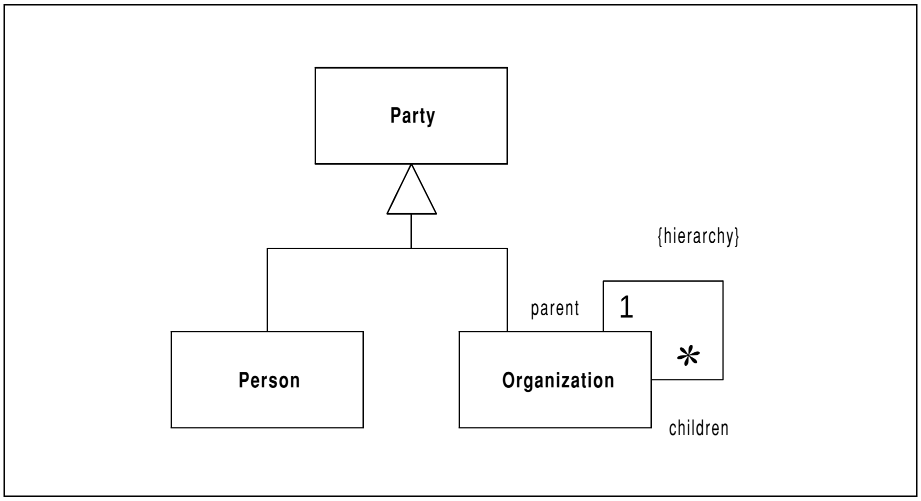

Figure 0.3 *Adding Party to an organization hierarchy*

Making a supertype for the organization is a pretty obvious move, another common, if less obvious, supertype is *Party (15)*: a supertype between the organization and person, leading to Figure 0.3. Often you find that there isn't much difference between the hierarchic association between organizations and the association between person and organization so you can pull these associations up to the supertype (Figure 0.4).

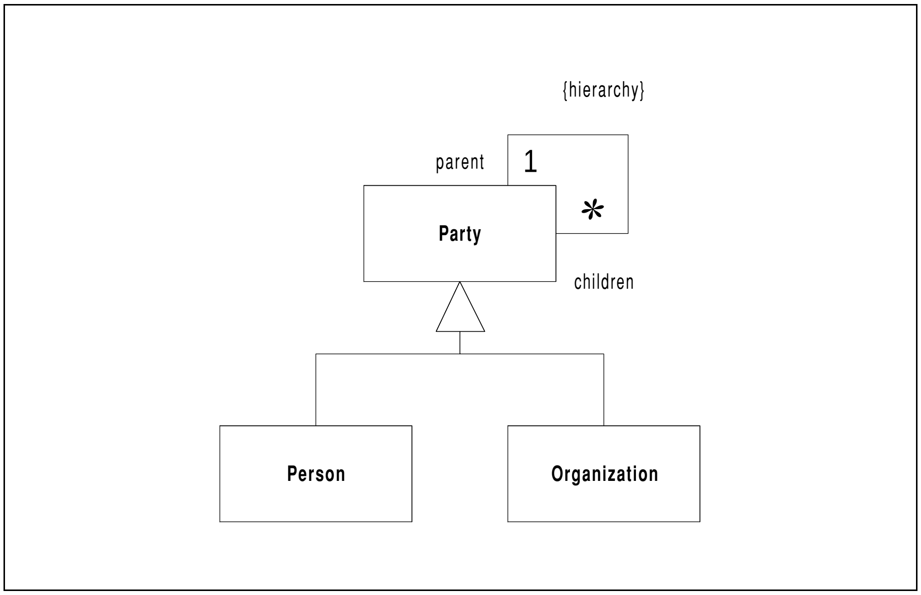

Figure 0.4 *A hierarchy on Party*

A hierarchy like this is a good choice for many organizations, it certainly captures the usual org charts pretty well. But as organizations get larger then you tend to see a number of different kinds of links between your parties. This might be matrix style organizational structures where people are organized by job function and geographic location at the same time. Obviously one way to do this is to create a second party hierarchy, but this only goes so far. You don't want your party festooned with hierarchies.

This situation leads you to *Accountability (27)*, where you make the interparty releationship an object in its own right, typed according to the kind of link you need to have (Figure 0.4). Accountabilities represent the most powerful, and also the most complex way of dealing with organizational strucutres. So like most of these power tools, you don't get them out unless you really need them. But when you do accountabilties give you a very flexible way of handling all sorts of relationships.

When you have accountabilties there are often rules that say what kinds of parties can be connected together. You can use a *Knowledge Level (42)* to capture and enforce these rules.

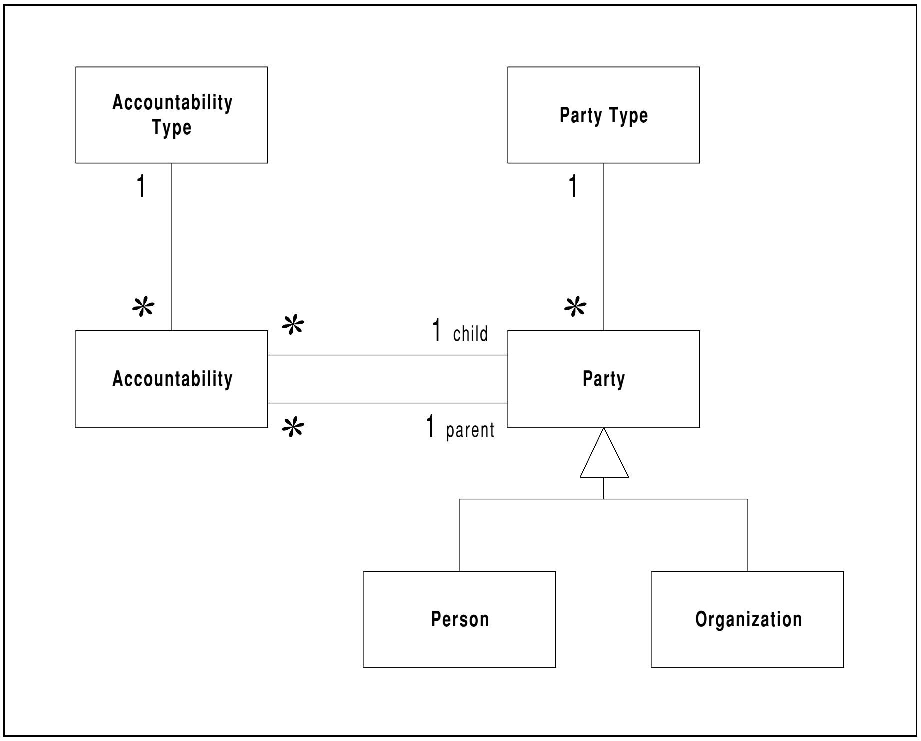

Figure 0.5 *Using Accountability for the organization structures*